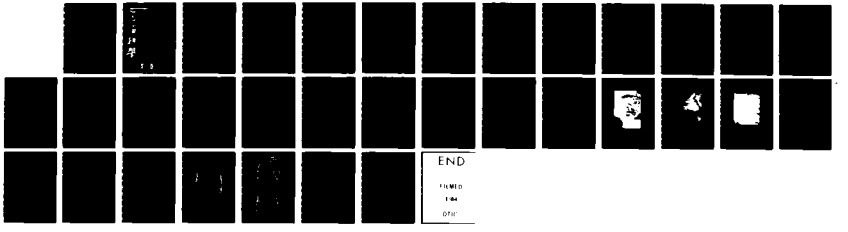
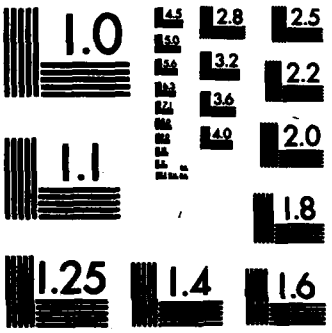
AD-A135 311 AN APPROACH TO KNOWLEDGE-DIRECTED IMAGE ANALYSIS(U) 1/1
ROCHESTER UNIV NY DEPT OF COMPUTER SCIENCE
D H BALLARD ET AL. SEP 77 TR-21 F30602-77-C-0050
UNCLASSIFIED F/G 20/6 NL

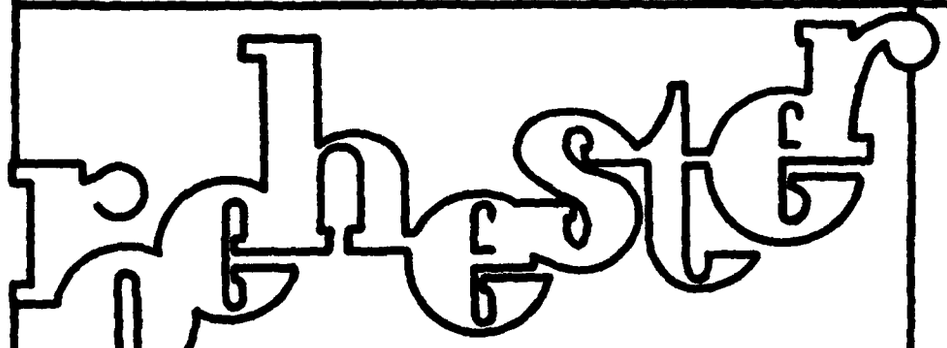




AN APPROACH TO KNOWLEDGE-DIRECTED IMAGE ANALYSIS

D.H. Ballard, C.M. Brown, J.A. Feldman
Computer Science Department
The University of Rochester
Rochester, New York

TR21
September 1977



Department of Computer Science
University of Rochester
Rochester, New York 14627

AN APPROACH TO

KNOWLEDGE-DIRECTED IMAGE ANALYSIS

D.H. Ballard, C.M. Brown, J.A. Feldman
Computer Science Department
The University of Rochester
Rochester, New York

TR21
September 1977

The preparation of this paper was supported by the Alfred P. Sloan Foundation under Grant No. 74-12-5, and by the Defense Advanced Research Projects Agency under Grant No. F30602-77-C-0050.

Document describes

Abstract

A vision system is described which uses a semantic network model and a distributed control structure to accomplish the image analysis process.

The process of "understanding an image" leads to the instantiation of a subset of the model, and the identification of nodes in the instance of the model with image features. The instantiated nodes and the relations between them form another data structure called the sketchmap. The sketchmap explicates the relation of the model to the image; this model-image mapping is accomplished by mapping procedures which are part of the procedural knowledge in the model.

The procedures are accompanied by descriptions which contain at least pre- and post-conditions for the procedure and performance measures for it. Nodes which have attached procedures may also have an executive procedure attached. This executive is responsible for deciding which of several possibly effective procedures to run. Thus through the executive the system does a very general kind of procedure invocation based not only on what the executive knows about global state, but on a rich description of the procedure's capabilities.

The user's program is generally responsible for allocating effort at a level above that of the individual executive procedure. Thus no single domain-independent formulation or methodology is imposed on all vision tasks. One facility provided by the system is the use of geometric constraints between model objects to guide search for the objects in the image.

The system is an attempt to bring together many current ideas in artificial intelligence and vision programming and thereby to cast some light on fundamental problems of computer perception. The semantic network facilitates the interplay between geometric and other relational constraints which are used to direct and limit search. The use of attached procedures in the network gives a mix of declarative and procedural knowledge, and the executive provides an unusually powerful procedure invocation scheme. The multiplicity of procedures allows modelling objects under radically different conditions and levels of detail. This tends to make the system robust in that an object which

could not be located initially may be found later when knowledge about the image has increased.

The system is illustrated throughout the chapter with illustrations from two particular applications: the finding of ships in a dock scene and the finding of ribs in a chest X-ray film.

1. Knowledge-Directed Image Analysis

Image analysis is the process of attaching meaning to an image. One way to do this is to use an explicit model of what the image can contain, and then construct a mapping between the model and the image. Since a particular image is only an instance of the class of images that the model was intended for, the model must be in some sense "larger" than the image. That is, a useful model of the domain of an image will typically contain a large amount of information on possible image content. However, the mapping generated by the analysis process (if we have a good model), will typically use only a small portion of this model.

An even smaller portion of the model is used in specialized analyses, for example, analysis performed to look for particular objects in the image. These processes typically require that only a small portion of the image be mapped onto a small relevant part of the model that explains that image. For example, in analyzing a radiograph for pneumoconiosis, we might use only a small portion of a radiograph model. We term a task which instantiates a subset of the model a query, to emphasize that only a portion of a large possible mapping is generated, and that we expect the working environment to be one consisting of a sequence of such tasks. Examples of a sequence of queries would be:

- returning to an aerial photo on different days to perform different tasks;
- different physicians requesting different differential diagnoses of a radiograph;
- generating different land use maps for agricultural and social scientists from ERTS data.

Given this approach to image analysis, we have defined a representation that allows for extensions of partial mappings which may be known a priori or acquired sequentially. Additionally, we have a way of defining quantitatively when the query has been satisfied so that we do not perform unnecessary mappings (e.g., the system may or may not need to know where all parts of an object are if it has "found" the object at some coarse level of detail). Thus the concept of a query is central to our approach to image analysis. Given a richly descriptive image model, our objective is to code a query to require mapping a minimum of model structure onto the image.

One of the problems in generating the mapping to satisfy a query is that the mapping is between very different structures. The natural elements of the model are objects which are represented symbolically, whereas the natural elements of the image are "picture elements," or pixels. To bridge this gap, we have structured our vision system in layers as shown in Figure 1. At the most abstract end of the structure is a semantic network representing our model. The model contains generic information encoded as idealized prototypes of structures from low level (such as edges) to high level (such as complex assemblages of objects in the world).

In the middle we have a sketchmap. This is a data structure that is synthesized during image analysis and provides associations between the model and the image, that is, the synthesized sketchmap is a network of nodes which turn out to be instantiations of a subset of model nodes. The need to differentiate between generic objects and instantiations of generic objects has been recognized in natural language understanding work, e.g. [Hayes, 1977]. We believe that it is also necessary for image analysis. Besides associations with the model, sketchmap nodes contain associations with image structures, such as edges and regions, specific to the particular image being analyzed.

At the other end of the vision system is a third structure termed the image data structure. This consists of the original image at different magnifications, spectra, resolutions, etc., together with various filtered versions of the image, texture images, edge images, etc. The parameters for generating all the image data structures are typically specialized to a particular model-image mapping context.

This multi-layered structure is reminiscent of the VISIONS System [Hanson and Riseman, 1977]. Both systems are designed to take advantage of variable resolution, both have a knowledge base or model of the world, a subset of which is mapped into the image. Perhaps the main difference is that in VISIONS, segmentation of the image into edges, regions, etc., is made to a level determined by the model so that the image will be understood to the fullest possible extent, given the knowledge in the model. In the system described here, the user's query is responsible for the level of detail the system pursues.

A successful analysis of an image contains two parts: the generation of the proper links between sketchmap nodes and image data structures and the generation of the proper links between sketchmap nodes and model nodes. We describe a procedure that generates the image-sketchmap link as a mapping procedure. A mapping procedure is a low-level procedure which is attached to a particular node in the sense of [Bobrow, 1977] and whose function is to refine the description of that node. The need for these kinds of procedures in image analysis has also been recognized by [Sloman, 1977]. The construction of links between the relational world model and the sketchmap is one of the functions of the executive procedure. The executive procedure embodies the overall strategy for achieving the goal(s) and is programmed in a high-level language (currently SAIL).

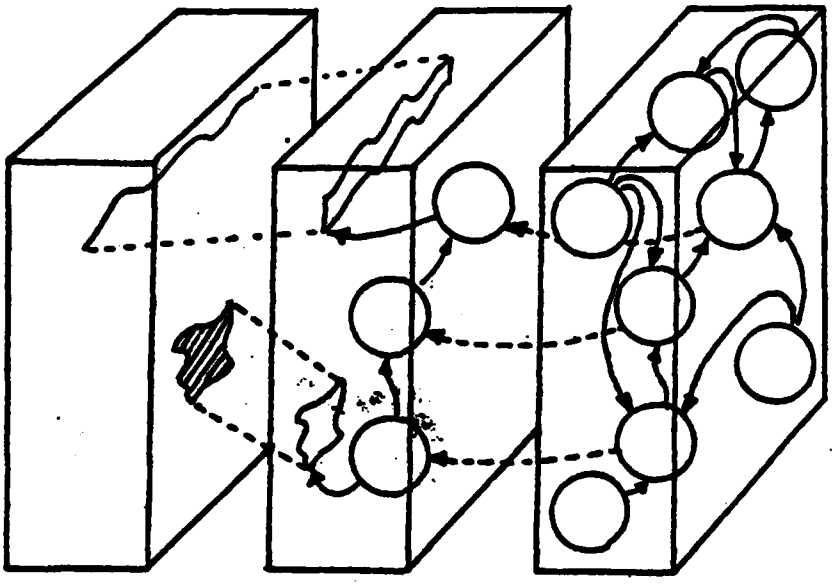


Figure 1. Basic Layer Structure

Since the best procedures for finding structures in real-world images are special-purpose, we have avoided imposing a uniform problem-solving regime on the mapping procedures; instead they must be coded especially to take advantage of the user's specialized knowledge of the domain. However, some program autonomy is allowed in the choice of mapping procedures. Where different mapping procedures can generate the same links between sketchmap nodes and image data structures, the executive procedure can select the most appropriate based on a description attached to each mapping procedure. Also, the executive procedure can use general geometric relational constraints to pin down the location of objects.

2. The Structure of the Model

2.1 The Concept of Template Nodes

The model holds different kinds of knowledge about the image domain. It includes a relational network of nodes which are identifiable with (primitive and complex) objects and concepts in the domain from which the scene is taken. The answer to a query is a synthesized sketchmap or description; this is an instantiation of a subset of the model. The model, therefore, contains knowledge in the form of all potential instantiable descriptions. An example of this kind of knowledge is the assertion:

"the sternum is above the heart."

This can be readily included in the model network. It is potentially part of the model-image mapping ("the sternum" and "the heart" could be instantiated with pointers to regions of the picture). The model also contains knowledge which is not in the form of a synthesized description but which, for instance, could be used in generating a description. An example of this kind of knowledge is:

"ships are about 6 times as long
as they are wide."

This knowledge can be included in the model network and may help a program that is searching for a ship, but will not become part of the model-image mapping. When it is meaningful to differentiate between these two kinds of knowledge, we will refer to the parts of the network that represent the former kind as template nodes. Synthesized descriptions will be directly related to these nodes and their arcs.

Each template node has a substructure which represents the sense in which that node is to be "understood." Prior to a query, the meaning of a node is defined in terms of a substructure of mapping procedures and constraint relations. After a query has been satisfied, the template node is represented by instantiated nodes with attached specialized location descriptions, as shown in Figure 2.1. Four basic types of links provide a simple syntax to the network structure. A powerful advantage to this syntax is that the executive procedure can direct the analysis in a more

general way, by using programs that function on classes of links representing different kinds of concepts, rather than on some set of specific links. The need for this organization in natural language understanding has been recognized by Brachman [1976].

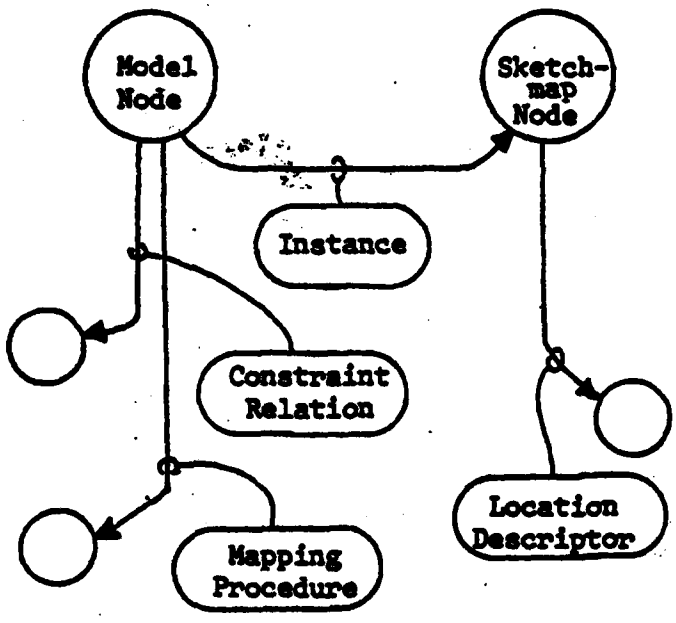


Figure 2.1 The Next Level of Detail in Model-Sketchmap Nodes

2.2 Constraints

Links to other model nodes encode (perhaps parametrized) constraint relations between model nodes. Links can encode:

- the probability that the relationship holds;
- a quantifier representing the expected value of the relationship.

For example, the relationship SHIP ADJACENT DOCK might have a certain probability of being true, and and an expected distance that the ship is from the dock. We refer to the template nodes and geometric relations between them as the constraint network. This network may be interpreted from two viewpoints. First, the existence of crucial constraint relations may be checked. This may be done through the matching features of the associative structures in SAIL. Secondly, if particular parameters arising from a constraint are needed, the network may be evaluated like a program to find subsets of the model or the image that satisfy the constraints. Its results take account of partial or unspecified information, and it may be updated upon receipt of better data with a minimal amount of work. It is much like the graph of variable dependencies in AL [Feldman et al., 1975]. In brief, each node has a "Constraint Operation," such as Intersection, Translation, Union, or indeed any function of up to two arguments; it has two operand nodes; a father node; a status that may be "Up-To-Date" or "Out-Of-Date"; and a value that is some data structure such as a number, a list of linear objects, a region, etc. Additionally, it may have information on how difficult the node is to evaluate, as a function of the contents of its operand nodes. This last feature allows some cost/benefit analysis of evaluation of sections of constraint networks.

The constraint network for the prose:

"The centroids of docked ships are on lines parallel to the intersection of coastlines with dock areas at a distance of one-half a ship width"

is shown in Figure 2.2.

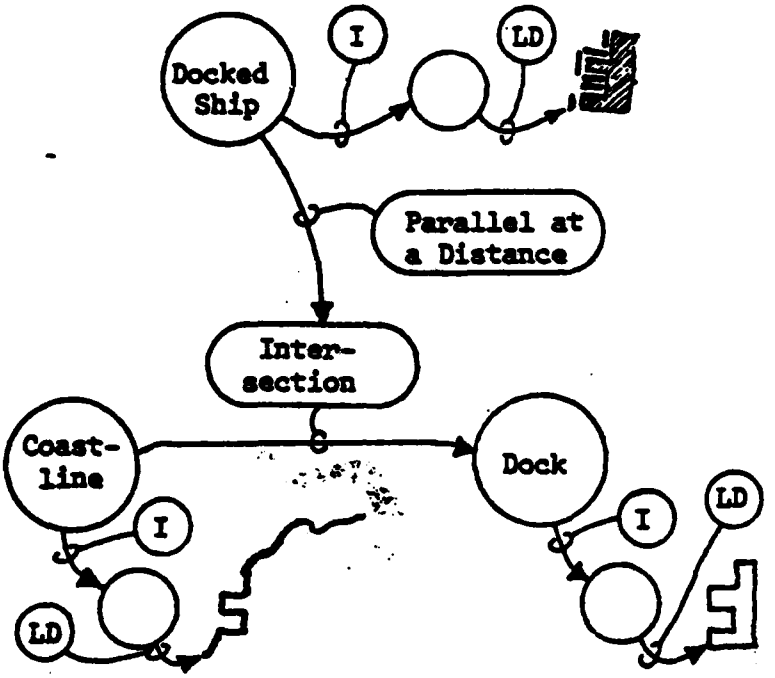


Figure 2.2 A Portion of a Constraint Network
(I = Instance, LD = Location Descriptor)

The network starts out with data (from the model or from previous scene analysis) as the values of the tip nodes, but no values at non-terminal nodes and all nonterminals marked Out-Of-Date. Data at a tip node can have one of three statuses: it can be known that the object does not exist in the scene (so the value of the node is the null set), it can be known to some degree of accuracy where objects are in the scene (so the value of the node is a subset of image or world points), or perhaps nothing is known (in which case the object could be anywhere, and the value is implicitly the universe of image or world points).

When the constraint network is evaluated to determine what is known about the location of its object, each node recursively evaluates its Out-Of-Date operand nodes, performs its operation, and stores the result in its value. It marks its status Up-To-Date. Intersection and Union work properly with the definitions of partial information of the last paragraph. When new (or better) information about an object at a tip of the network comes in, all nodes on a path from the tip to the root are marked Out-Of-Date. Then when the network is next evaluated, (only) the necessary partial results are re-computed. In keeping with our philosophy, the network is not self-activating, but is run on explicit user command.

2.3 Location Descriptors

A location descriptor provides information about where to find an entity. The part of the location descriptor which specifies a point set enclosing the region has been referred to as a tolerance region [Bolles, 1975]. A shape location descriptor might have the structure shown in Figure 2.3.

```
[ ShapeLocationDescriptor
    nodetype:     specialization prototype
    instance-of:  a LocationDescriptor
    locates:      OneOf {(a ShapeObject),
                         (a ShapeFeature)}
    coordsystem:  a CoordinateSystem
    centroid:     a PointSet
      //allows for "fuzziness"
    orientation:  an AngleRange
      //...ditto
    tol. region:  a PointSet ]

    ...similarly for Point, Linear, and
      AreaLocationDescriptors

[ CoordinateSystem
    nodetype:     abstract prototype
    units:        a LengthUnitSpecification
    scale:        a NumberRange
      //length units / system unit
    transforms:   SetOf {((a Coordinate  Transform)
                          (a Coordinate System)),
                         ...} ]
```

Figure 2.3 Example of a Shape Location Descriptor

This organization is suggestive of a frame-like structure [Minsky, 1975]. However, not all the entries need exist; just the syntax is necessary to allow the entries to be found. In practice only the properties relevant to a particular query will be generated. Such partial instantiations are easy in the SAIL associative structures [Feldman and Rovner, 1969].

Our examples of geometrical constraints, locations, etc., are two-dimensional. There is a large class of interesting images that are inherently two-dimensional (ERTS images, light or transmission electron microphotographs, CAT scanner images, bio-ultrasound images) as well as some that for some purposes may be treated as two-dimensional (aerial reconnaisance imagery, medical X-ray imagery, natural scenes, etc.). Of course it is often helpful to know about 3-D when processing natural scenes [Garvey, 1976], and it has been demonstrated that a 3-D model of the world is necessary to accomplish some tasks with aerial mapping photographs taken from 35,000 feet [Barrow, 1977]. Within the framework of the system described here, 3-D world coordinate systems would be linked through camera-transformation coordinate transforms to image coordinate systems. The location descriptors would be in terms of the relevant coordinate systems.

There are many advantages to having a standard representation for object locations:

a. If such descriptions are data types, their computations can be separated from the procedures that use them. If they can be passed as arguments, they provide a certain "common currency" between procedures, thus simplifying and modularizing the procedures that use them.

b. Location descriptors can represent approximate locations, which is useful for queries unconcerned with exact answers.

c. Constraints between locations can propagate knowledge throughout the model. Location descriptors can be computed from other location descriptors via relations, or by union and intersection of the described point sets. A system which applied linear programming techniques to the problem of locating regions through constraints placed on their boundaries was developed in [Taylor, 1976].

d. Use of location descriptors is geared to an abandonment of the exhaustive segmentation paradigm wherein every region must correspond to some object. Different location descriptors may refer to disjoint point sets or may overlap on the image.

3. Control

3.1 General Philosophy

Generally a query results in the synthesis of a sketchmap with instance nodes whose location descriptors are accurate enough for the purposes of the query. A query might also result in further refinement of location descriptors of the extension of an existing sketchmap to account for more image structure. A query-directed vision system should thus be able to use relevant information (i.e., the state of the analysis) generated in successive queries. Most queries will take the form of user-written executive programs, since nontrivial tasks usually require fairly rigid recommendations about how the system should go about solving them. Initially the system will not attack the problem of automatically translating queries in some command language into executive programs.

Figure 3.1 shows the SAIL code used in a very simple executive procedure for selecting mapping procedures which identify instances of rib nodes in chest radiograph images. Each mapping procedure has pre-conditions, including an associated accuracy measure, which can depend on its neighbors, as well as a cost measure. The cheapest rib procedure which satisfies the pre-conditions is selected. Each rib node is searched for once and there is no facility for dealing with failures or mistakes. But the important point here is that the executive can have a relatively simple structure. This facilitates experimentation with various control strategies other than the depth-first strategy shown in the example.

3.2 Characterizations of Mapping Procedures

Mapping procedures have associated descriptions which are used by executive procedures. The descriptions contain the following:

- the slots in the data object which must be filled for the procedure to run;
- the slots the procedure can fill in;
- the cost and accuracy of the procedure in some meaningful units;
- the a priori reliability of the procedure.

Some rib mapping procedure descriptions are shown in Table 4.1, but these do not tax our representation scheme. More difficult examples of the kinds of facts we expect to be able to encode in this structure are (for a straight-line structure) that a Hough transform [Duda and Hart, 1972] cannot find the endpoints of a line but is more reliable than the cheaper Shirai tracker [Shirai, 1975], which itself needs to know the direction of a line before it can track it, and that a Heuckel operator [Heuckel, 1971] is more expensive, but can furnish many facts about the line with little known a priori, and can rate itself on reliability of its result.

There are several advantages to separating the executive procedure from the mapping procedures and their descriptions:

a. The executive procedure can be written more easily without considering the implementation details of mapping procedures in great depth.

b. Mapping procedures are similarly simplified without the burden of determining an appropriate context for their application [Sloman, 1977].

c. The executive procedure can automatically select alternative procedures in the event of mapping procedure failures.

d. Descriptions allow a choice between methods (if several are available) based on capability, resource requirements, and a priori reliability. (Also, recovery from failure of individual routines can be automated through planning [Feldman and Sproull, 1975].)

e. If the mapping procedures can produce reliable a priori estimates of their success the analytical results of [Bolles, 1975] and [Taylor, 1976] could be extended to select the procedure which produces sufficiently exact data objects.

```
Recursive Procedure MatchRib(itemvar Node);

begin
  itemvar x, v; integer Var;
  if INSTANCE of Node is ANY
  then
  begin
    Print("rib ", Node, " already matched");
    return;
  end
  else
  ! find and run procedure to do job at min cost;
  begin
    itemvar TempProc; integer MinCost, TempCost;
    MinCost := VeryLarge;
    foreach x such that
      RIB!PROCEDURE of Node is x do
      begin
        Var := GetConstraintsAndVariance(Node, x);
        if Var < Tolerance
        then
        begin
          TempCost := FindCost(Node, x);
          if TempCost < MinCost
          then
          begin
            TempProc := x;
            MinCost := TempCost;
          end;
        end;
      end;
    if MinCost = VeryLarge
    then
      Print("No proc. can do job for rib ", Node)
    else ApplyProc(TempProc, Node);
    foreach v such that NEIGHBOR of Node is v
                    and TYPE of v is RIB
                     do MatchRib(v);
  end;
end;
```

Figure 3.1 Executive Procedure for Ribs

4. Applications

4.1 Finding Docked Ships

Finding ships in a dock scene illustrates how high-level metrical knowledge about the image (such as provided by a topographic map) can make certain scene analysis problems easy.

The model contains, in a Constraint Graph form (see Section 2.2), the knowledge that docked ships are in the ocean adjacent to dock areas, parallel to the dock and with a centroid a distance away related to the width of the ship. In a Shape Object Descriptor, some facts about the sorts of ships we are trying to find are stored, viz., a template for matching them (in our case, a rectangle of 1's in an array for template-matching), their width, length, average brightness, etc. Template-matching is among the simplest vision primitives. Only in a context having a great deal of structure could it be expected to work in scenes as complex as Figure 4.1a.

Figure 4.1a is from a USGS mapping photograph. It roughly corresponds to the topographic map of Figure 4.1b. Included in the map are such linear features as coastlines and dock areas. From the digitized photo, a small (196 x 164) window is extracted and stored on disk. A half-toned version of this window is shown in Figure 4.1c. From the map, the coastline and a dock area are extracted and stored on disk; this information is shown in Figure 4.1d. Map information may be automatically registered with photographic images to high accuracies by techniques developed at SRI [Barrow et al., 1977]. For our study the registration was performed manually.

The system, under direction of the user-written query, begins by deciding where to look by satisfying a constraint network; the more information provided, the narrower the focus of attention. In the case illustrated in this section, the constraint network looks as it does in Section 2.2. Presupposition of "perfect" registration leads to sharp lines of search specifying loci of ship centers. Imperfect registration would give fuzzier loci.

The linear loci and the orientational constraint on the ships means a simple template-matching technique will suffice to do the ship-finding job efficiently. (In this exercise it was the only technique, but an executive procedure might well have chosen it as applicable.) The ship template is rotated to be parallel to the midline as given by the constraint graph, and template-matching is done along the line; note is taken of where the score for the match goes over threshold, and when it comes back down under threshold. The average of these two positions is taken as the location of a ship. The black squares in Figure 4.1d show the results.

Our USGS mapping photograph is digitized to 256 grey levels on a .007" grid. The image is stored on disk with comprehensive and expandable header information. The image may be windowed and sampled at integral size reductions into an integer array in core for processing.

The system has representations for linear objects and regions. Linear objects are SAIL records making linked lists of (x,y) points. They can have three types at present: a list of points to be connected in order; a list of segments, i.e., pairs of endpoints to be connected pairwise; and logically circular lists of points representing boundaries. A robust and general routine based on merging was written to compute the intersection of such linear features. Other useful geometric routines find the distance of a point from a segment (not a line), and compute a segment parallel to and some distance from another segment.

Regions (except for templates, which are arrays) are SAIL list items. A region is a list of y-lists; a y-list has a y-value followed by an even number of x-values. The first x-value is an "entering region" boundary point, the second is a "leaving region" boundary point, and so on alternately. The region:

```
001
101
011
```

would be represented as

((1 2 3)(2 1 1 3 3)(3 3 3)).

Routines were written, again based on merging, to create the union and intersection of such regions, and to convert (via an asymmetric DDA algorithm [Newman and Sproull, 1973]) linear objects to regions. We find multiple representations of objects simplifies the work of routines such as the constraint primitives.

Template-matching utilities can produce an array containing a rotated and scaled version of a template and can compute the correlation of a template (at some rotation and translation) with the image array.

4.2 Finding Ribs in Chest Radiographs

The problem of finding ribs in chest radiographs illustrates the use of multiple procedures attached to the same template node (cf. Section 3.1). It uses the less precise geometric constraints arising from anatomy rather than cartography.

The model contains nine right and left ribs (the maximum amount normally visible on a chest film). Presently only the lower edge of each rib is detected. Each rib is modelled as a template node with offset parameters from itself to each immediate neighbor (above, below, opposite). Additionally, three different mapping procedures are attached to each rib node as shown in Table 4.1.

LookForARib uses the Wechsler parabolic model [Wechsler and Sklansky, 1975] to find a rib segment. AffirmARib translates that segment using the offset parameters and attempts to verify the presence of a rib by a correlation technique. HallucinateARib instantiates a rib by translating a neighbor with no verification.

Table 4.1 RibFinding Mapping Procedures

Procedure	Precondi-tions	Cost	Var.	Postcondi-tions
LookForARib	none	20	0	instance of rib
AffirmARib	instance of neigh-bor in sketchmap	4	1	instance of rib
HallucinateARib	instance of neigh-bor in sketchmap	1	5	instance of rib

Figure 4.2 shows a trace of the display during the rib-finding process. For this trace a slightly more complex executive than the one shown in Figure 3.1 was used. If a mapping procedure failed, another was chosen from the remaining applicable set. In Figure 4.2a large rectangles enclosing the lung fields have been found (by a lung query executive) and the smaller rectangles are plans for LookForARib, which is the only mapping procedure that can be applied. The horizontally-oriented rectangle defines an area to look for rib edges for the model node RIGHTRIB4 and the vertically-oriented rectangle defines an area for foci of a parabola representing the rib border. Figure 4.2b shows the resultant rib found. Figure 4.2c shows the plan derived from the constraints for the opposite rib, LEFTRIB4. Note that the plan now has the shape of RIGHTRIB4. Figure 4.2d shows the instantiation of LEFTRIB4 found by AffirmARib. Figure 4.2e shows the next two ribs found and Figure 4.2f shows the entire set of ribs. The ribs marked with the box (□) are found by HallucinateARib, due to the failure of AffirmARib. AffirmARib fails when the edge data is extremely poor.

To appreciate that the ribs found by the rib executive actually match the edge data, compare Figure 4.3a with 4.3b, which shows the results from another chest radiograph. Figure 4.3a shows the principal edges in the image and the latter has the ribs overlaid on top of those edges.

5. Summary

The semantic network is a kind of lumped parameter model in the spirit of [Fischler and Eschlager, 1973]. The geometric constraints in the network relate template nodes whose descriptions (the "lumped parameters") are generated by attached mapping procedures. The key difference is that information found during the analysis can change the way template nodes are located.

In analyzing an image it is crucial that the generating of abstract descriptions of parts of the image (i.e., segmentation) be intimately connected with the interpretation of those parts. In our system the former operation corresponds to generating sketchmap-image links whereas the latter corresponds to generating model-sketchmap links. Interpretation and segmentation are united through multiple mapping procedures and the executive, which can efficiently change the way a part of the image is analyzed as new information about the rest of the image develops.

Finally, we want the image analysis process to do as little work as possible to satisfy a given task or query. This is attempted through the specialization of all parameters to the given task, the inclusion of performance and accuracy measures in the mapping procedure descriptions, and the use of the constraint network. All of this is just the beginning of a long-term effort to study what can be done in a general way for goal-directed image understanding tasks.

Figure 4.1a Aerial Mapping Photograph

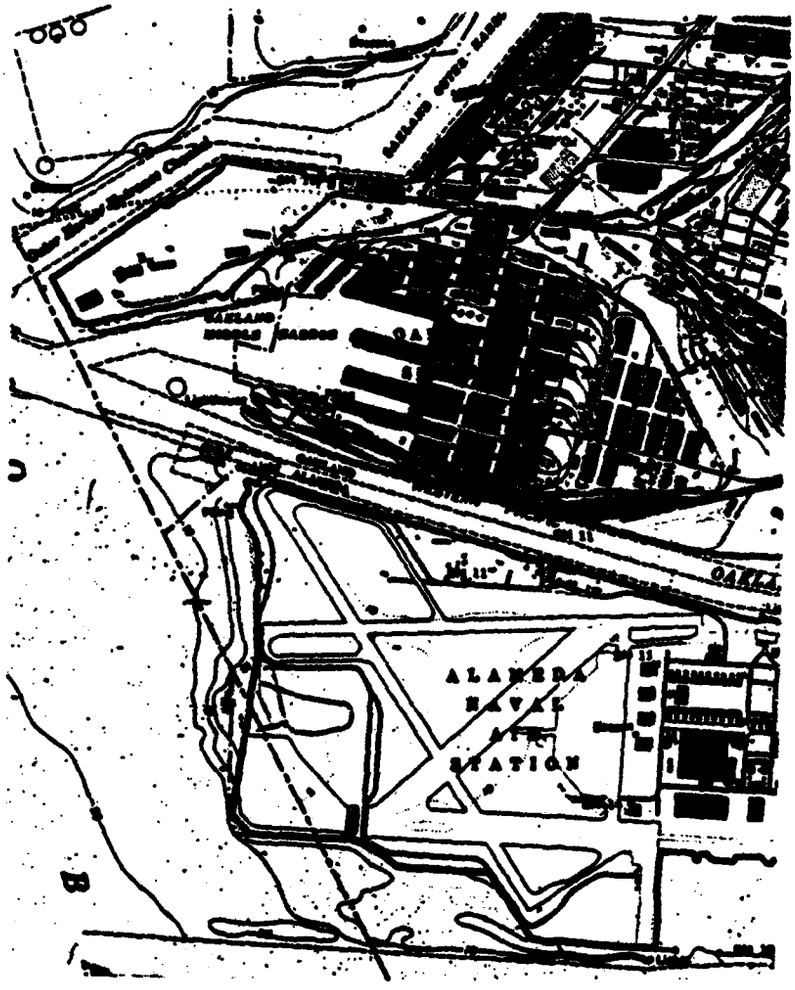


Figure 4.1b Topographic Map of Area in 4.1a.

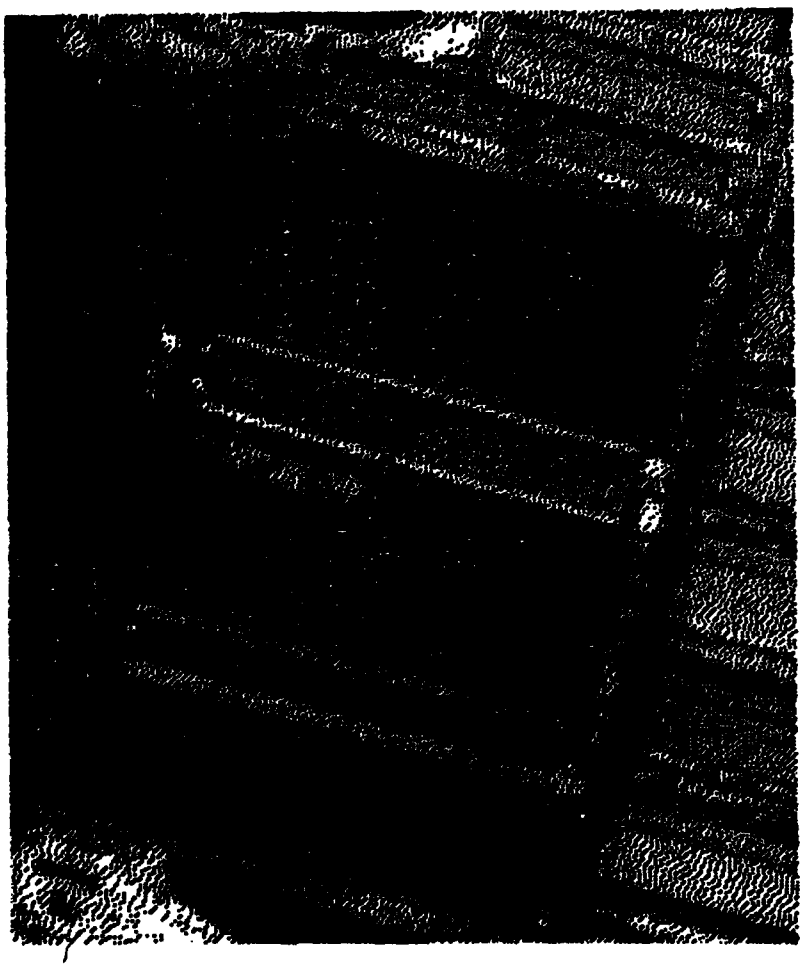


Figure 4.1c Halftoned Representation of Window from Digitized Version of 4.1a.

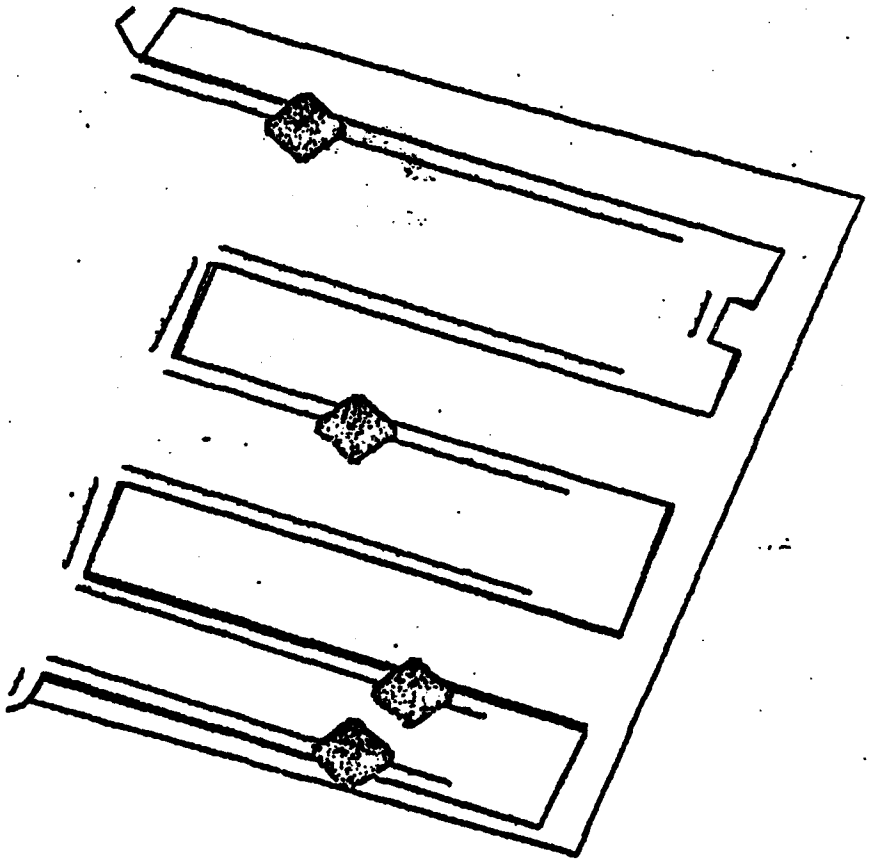


Figure 4.1d Coastline, Dock Area, Loci of Possible Ship Centers, Points of Application of Ship Template and Location of Ships.

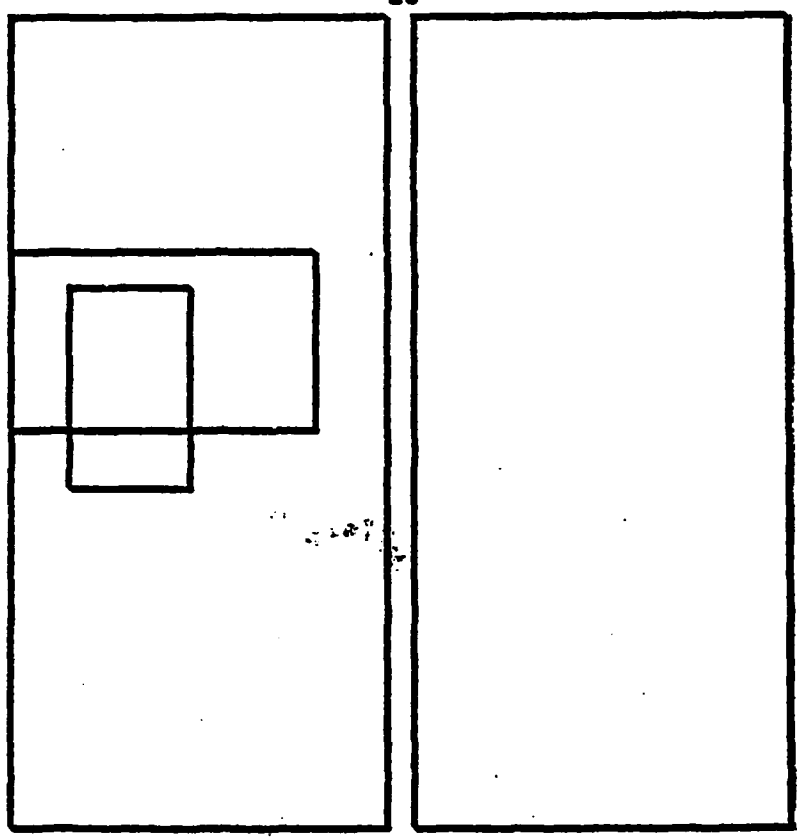


Figure 4.2a Lung Boundaries and Plan for RIGHTRIB4

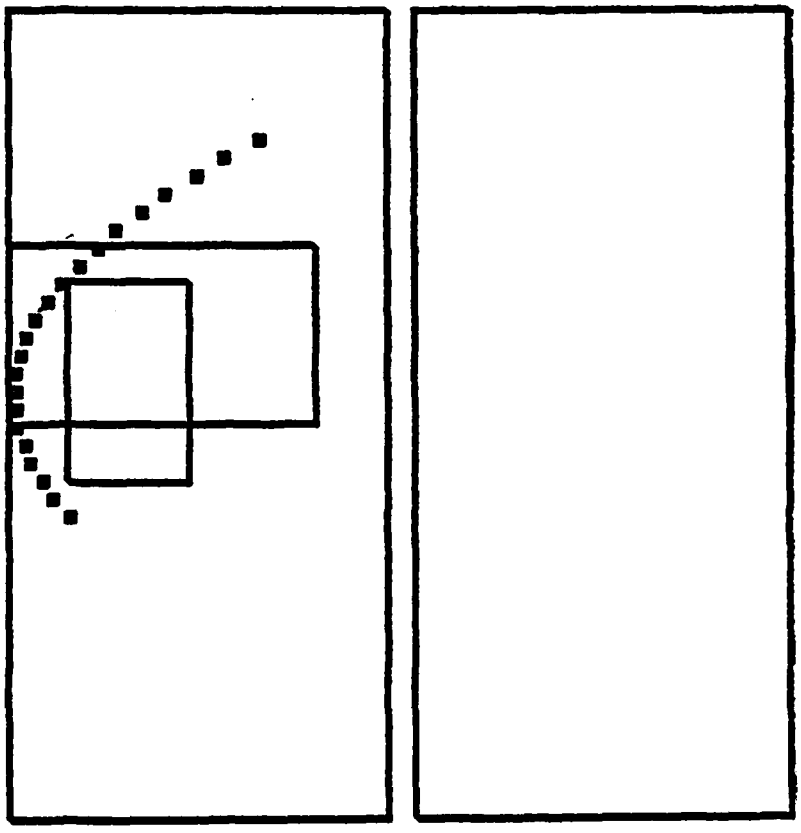


Figure 4.2b RIGHTRIB4 Found

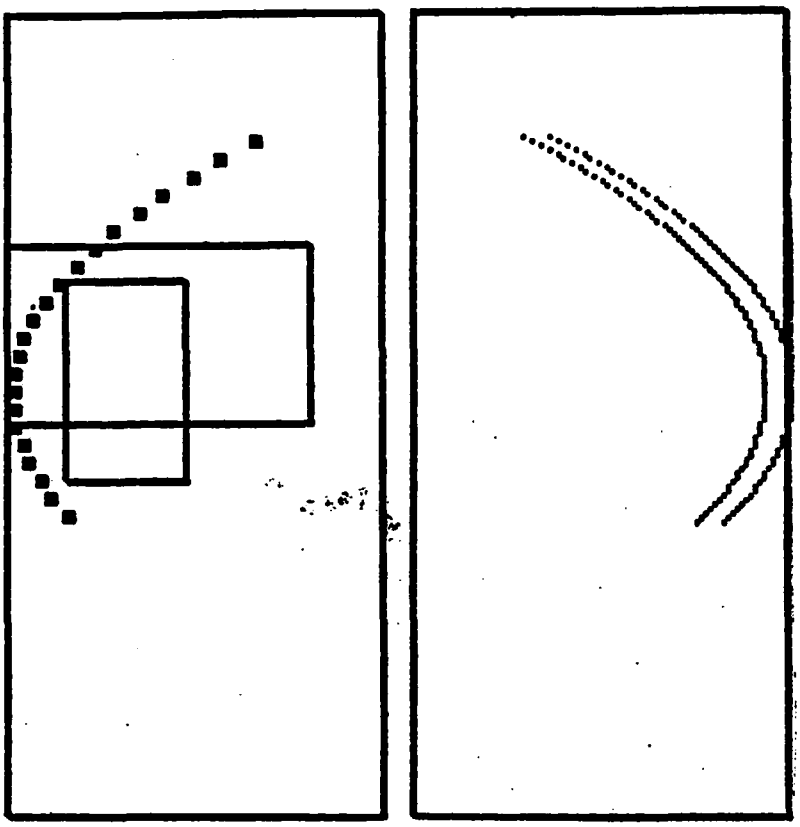


Figure 4.2c Plan for LEFTRIB4

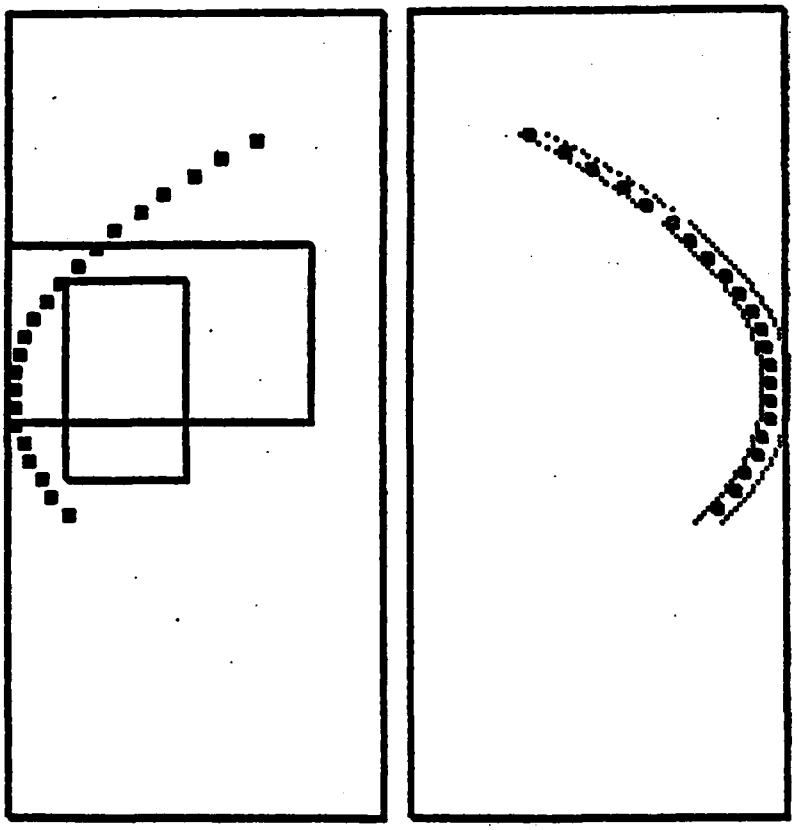


Figure 4.2d LEFTRIB4 Found

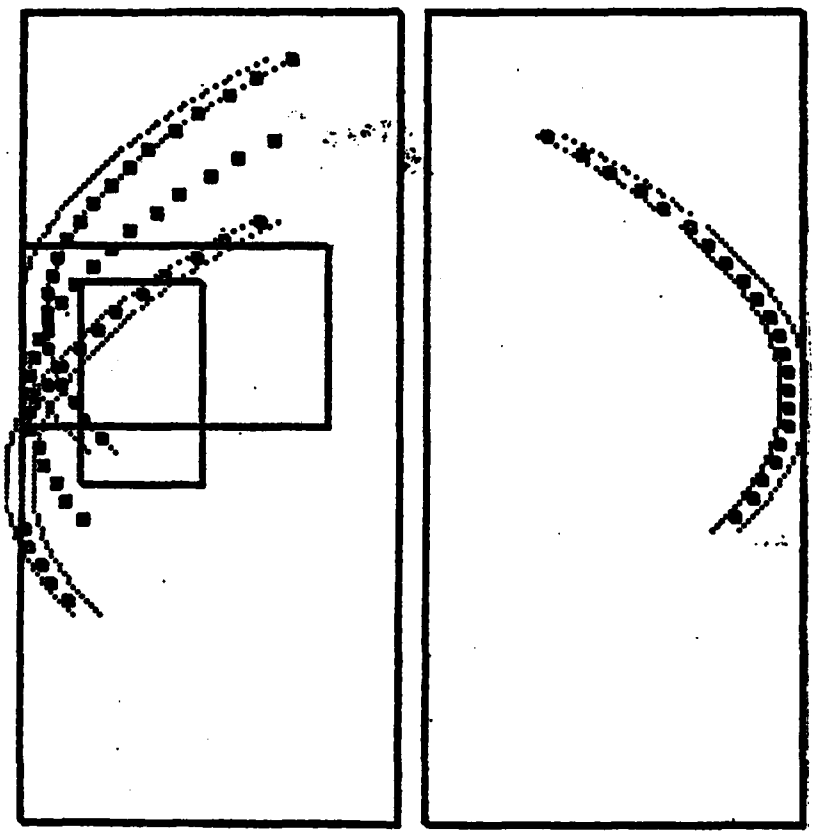


Figure 4.2e RIGHTRIB3 and RIGHTRIB5 Found

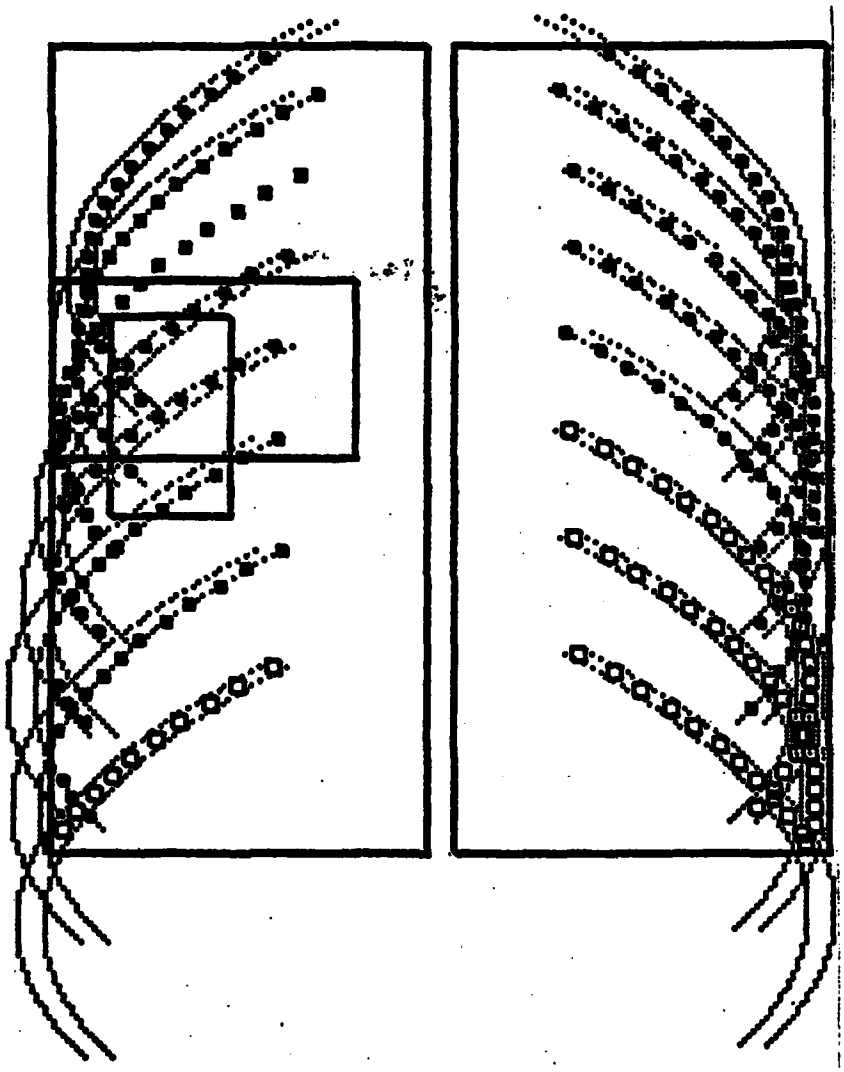


Figure 4.2f Final Result

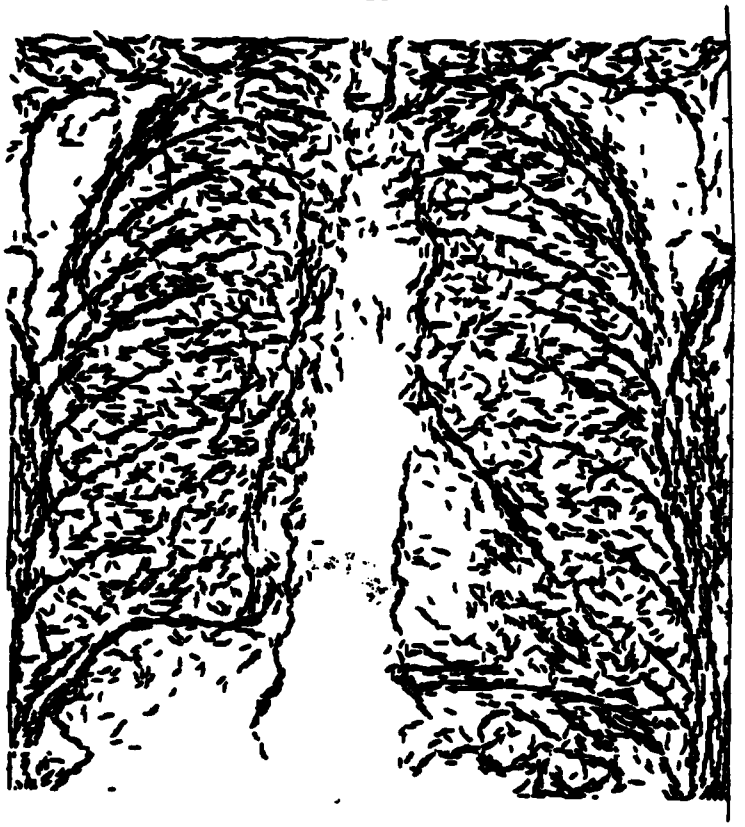


Figure 4.3a Edges Detected by the Heuckel Operator

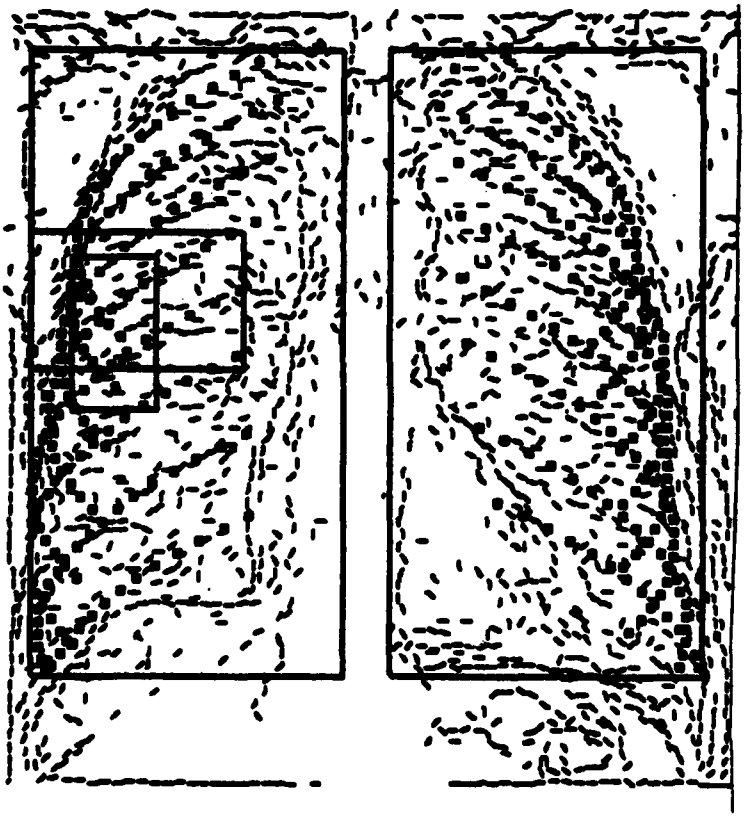


Figure 4.3b The Overlay of Rib Borders

References

Barrow, H.G. Interactive Aids for Cartography and Photo Interpretation SRI, Semi-Annual Technical Report, November 1976 - May 1977.

Barrow, H.G., Tenenbaum, J.M., Bolles, R.C., and Wolf, H.C. Parametric Correspondence and Chamfer Matching; Two New Techniques for Image Matching, DARPA Conference, Minneapolis, MN, April 1977.

Bobrow, D.G. and Winograd, T. Experience with KRL-0: One Cycle of a Knowledge Representation Language, Proc. IJCAI5, M.I.T., August 1977.

Bolles, R. Verification Vision Within a Programmable Assembly System, Stanford AI Memo, AIM-275, December 1975.

Brachman, R.J. What's in a Concept: Structural Foundations for Semantic Networks, BBN REP #3343, October 1976.

Duda, R.O. and Hart, P.E. The Use of the Hough Transformation to Detect Lines and Curves in Pictures, Comm. ACM, Vol. 15, January 1972.

Feldman, J.A., Finkel, R., Taylor, R., Bolles, R., and Paul, R. An Overview of AL, A Programming System for Automation, Proc. IJCAI4, Tbilisi, U.S.S.R., 1975.

Feldman, J.A. and Rovner, P.D. An Algol-Based Associative Language, Comm. ACM, Vol. 12, No. 8, August 1969, pp. 439-449.

Feldman, J.A. and Sproull, R.F. Decision Theory and Artificial Intelligence II: The Hungry Monkey, TR2, U. of Rochester, Computer Science Dept., Rochester, NY, November 1975.

Fischler, M.A. and Eschlager, R.A. The Representation and Matching of Pictoral Patterns, IEEE Trans. on Computers, C-22, January 1973.

Garvey, T.D. Perceptual Strategies for Purposive Vision, SRI AI Center, Tech. Note 117, September 1976.

Hanson, A.R. and Riseman, E.M. A Progress Report on VISIONS: Representation and Control in the Construction of Visual Models, COINS Tech. Report 76-9, July 1976.

Hayes, P.J. On Semantic Nets, Frames and Associations, TR19, U. of Rochester, Computer Science Dept., Rochester, NY, August 1977; also Proc. IJCAI5, M.I.T., August 1977.

Heuckel, M.H. An Operator which Locates Edges in Digitized Pictures, J. ACM, Vol. 18, No. 1, January 1971.

Minsky, M. A Framework for Representing Knowledge in Winston (Ed.), The Psychology of Computer Vision, New York, McGraw-Hill, 1975.

Newman, W.M. and Sproull, R.F. Principles of Interactive Computer Graphics, McGraw-Hill, 1973.

Shirai, Y. Analyzing Intensity Arrays using Knowledge about Scenes, in Winston (Ed.), The Psychology of Computer Vision, McGraw-Hill, 1975.

Sloman, A., et al. Popeye's Progress Through a Picture, Cognitive Studies Programme, School of Social Sciences, U. of Sussex, March 1977.

Taylor, R.H. Generating AI Programs from High-Level Task Descriptions, Ph.D. thesis, Stanford AI Lab, 1976.

Wechsler, H. and Sklansky, J. Automatic Detection of Rib Contours in Chest Radiographs, Proc. IJCAI4, Tbilisi, U.S.S.R., 1975.

END

FILMED

1-84

DTIC